REMARKS

I. INTRODUCTION

In response to the Office Action dated November 18, 2004, claims 1, 17, and 20 have been amended. Claims 1-6 and 8-20 remain in the application. Entry of these amendments, and re-consideration of the application, as amended, is requested.

II. CLAIM OBJECTIONS

In paragraph (1) of the Office Action, claim 6 was objected to for not ending in a period. Applicants note that the prior amendment did not delete the period. Accordingly, no amendment to claim 6 is necessary. In this regard, Applicants respectfully request withdrawal of the objection.

III. PRIOR ART REJECTIONS

In paragraphs (1)-(2) of the Office Action, claims 1-6, 8-13, and 15-20 were rejected under 35 U.S.C. §102(b) as being anticipated by Bardasz, U.S. Patent No. 5,689,711 (Bardasz). In paragraphs (4)-(5) of the Office Action, claim 14 was rejected under 35 U.S.C. §103(a) as being unpatentable over Bardasz in view of Applicants' Admitted Prior Art (APA).

Applicants respectfully traverse these rejections.

Specifically, the independent claims were rejected as follows:

> As to claim 1, BARDASZ teaches a method for providing access to application data items (arguments / values) of an application program (computer program / editor / visual programming environment) (col. 7, lines 59-66; col. 2, lines 41-45), the application data items (arguments / values) being contained in a plurality of interconnected data objects (data objects) processed by the application program (col. 6, lines 42-59), the method comprising the steps of: an extension object (operator object), that is associated with one of the plurality of data objects (data objects), receiving a request (operation / evaluation) related to at least one of the application data items (arguments / values), the request referring to the associated data object of the plurality of data objects; the extension object (operator) fulfilling the request with respect to the data object referred to by the request by creating a data provider object (relation object) (via the API processor) that is configured to provide access to internal data of the associated data object (data object); and if the request concerns at least one other data object (another data object) of the plurality of data objects, the extension object forwarding the request to one or more additional extension objects (operator objects) that are each associated with other data objects (data objects) for further processing of the request (via the result of the evaluated operator may serve as an argument to other operators) (col. 6, lines 42-63; col. 9, lines 25-31; col. 10, line 56 – col. 11, line 5; col. 11, line 64 – col. 12, line 37).
>
> As to claims 17-19, reference is made to a computer program product that corresponds to the method of claims 1, 2 and 15 and is therefore met by the rejection of claims 1, 2, and 15 above.
>
> As to claim 20, reference is made to an apparatus that corresponds to the method of claim 1 and is therefore met by the rejection of claim 1 above.

Applicants traverse the above rejections for one or more of the following reasons:

(1) Neither Bardasz nor APA teach, disclose or suggest an extension object that creates a data provider object;

(2) Neither Bardasz nor APA teach, disclose or suggest a data provider object that is configured with functionality to access to internal data of an associated data object;

(3) Neither Bardasz nor APA teach, disclose or suggest a data provider object that is configured with functionality to expose internal data of an associated data object; and

(4) Neither Bardasz nor APA teach, disclose or suggest an extension object forwarding a request to additional extension objects that are associated with other data objects for processing of the request.

Independent claims 1, 11, and 20 are generally directed to providing access to application data items of data objects. Specifically, an extension object is associated with a data object (that contains the data items). The extension object receives a request related to one of the data items. In this regard, the request merely refers to the data object that is associated with the extension object. The extension object fulfills the request by creating a data provider object that is configured with various functionality. Specifically, the claims provide that the functionality provides access to internal data of the associated data object (i.e., the data provider provides access to the data items) and exposes the internal data of the associated data object. Additionally, if the request concerns at least one other data object, the extension object forwards the request to additional extension objects that are associated with the other data objects.

In view of the above, the claims set forth three specific types of objects: data objects, extension objects, and data provider objects. The data objects contain internal data referred to as data items. Each extension object is associated with a particular data object. The extension object is configured to create a data provider object for the associated data object. The data provider object has functionality to provide access to the internal information of the data object (i.e., the data item) and to expose the internal information of the data object. None of the cited references teach such claim elements.

Bardasz merely describes a method and apparatus for converting a set of functions of any software system that does modeling into a corresponding set of parametric functions that, when

called, generate not only a resulting model, but also a dependency graph representation of the model. The dependency graph can include directed functions and non-directed constraint relationships, and is used when a change is made to the model so that only affected portions of the model are reevaluated. The dependency graph can be visually presented to a user, and can be created or edited through a visual programming environment. The graph can be modified to change either input values to the model, or the graph elements that represent the functions used in creating the model. When changes to the dependency graph are made, it can be reevaluated to incorporate the changes into the model. When the visual programming environment is used to modify the graph, the environment calls the parametric set of functions. The set of parametric functions is generated by creating wrappers in a standard programming language around the functions of the software system. The dependency graph can also be used to generate a computer program in a standard language that, when executed on the software system, regenerates the model as well as the dependency representation. The computer program can also be edited and then executed on the software system to generate a modified version of the model.

The Office Action rejects each of the claimed objects based on particular elements in Bardasz. Specifically, the claimed extension object was rejected based on Bardasz' operator object. The claimed data object was rejected based on Bardasz' data objects. And the claimed data provider object was rejected based on the relation object. Applicants respectfully traverse such rejections.

Firstly, the claimed extension object receives a request relating to at least one of the data items by referring to a particular data object. Bardasz' operator object does not receive any request. Instead, Bardasz' operator is a component that takes action on at least one data object and is a function or evaluable expression that represents the design actions taken by the user (see col. 6, lines 50-53). However, note that Bardasz' operator does not receive a request that refers to a particular data object. Instead, Bardasz' operator is a command itself and does not receive a request.

Secondly, the claimed extension object fulfills the request by creating a data provider object. Applicants submit that Bardasz' operator does not create any such data provider object. To read on the claims, Bardasz' operator must create the relation. Such a creation by the operator does not occur. Instead, Bardasz' relation is merely a component that identifies what role each data object plays with respect to an associated operator. There is no suggestion, implicit or explicit, that Bardasz' operator is the object that creates the relation. The Office Action asserts that the creation

occurs via the API processor. However, Applicants note that whether it occurs through the API processor or nor, Bardasz' operator is not the entity that creates the relation. In this regard, the claims are specific and state that the extension object fulfills the request by creating the data provider object. Thus, the claimed extension object does the creation in order to fulfill a received request. Bardasz' operator has nothing to do with the creation of the relation. Further, the relation is not created to fulfill the request. Instead, the API processor creates the various entities including an entity for each operator that includes a list of pointers to graph entities that represent its relations. Such a creation is not conducted by the operator through the API. Instead, such a creation is independent of the operator and is not by or pursuant to the control of the operator.

Thirdly, the claimed data provider object is configured with functionality to provide access to the internal data and to expose the internal data of the data object. To read on (or render obvious) the claims, Bardasz' relation (which is equated to the data provider object in the Office Action) must have functionality that provides access to and exposes the data. However, Bardasz' relation does nothing of the sort. Instead, as described above, Bardasz' relation merely identifies what role each data object plays with respect to an associated operator (see col. 6, lines 54-56). Further, the relations define the relationship of a data object to an operator (see col. 9, lines 25-29). However, such a relationship between a data object and an operator does not even remotely describe functionality in the relation that provides access to the data in the data object and that exposes the data in the data object. In this regard, Bardasz' relationship between two objects (i.e., the relation object) is not configured to provide access to any data. Instead, it merely establishes the relationship between Bardasz' operator and data object. Such a relationship does not provide access to the data in the data object. Further, such a relation does not expose the data in the data object.

Applicants note that Bardasz' data object itself has the pointer to the memory location wherein the argument value is stored (see col. 11, lines 1-5). Accordingly, rather than using a relation to provide access and expose such information, Bardasz teaches the data object itself maintaining and providing access to the information. Such a teaching is clearly distinguishable from and does not render obvious the presently claimed invention. Instead, such a teaching teaches away from the present invention.

Lastly, the claims specifically provide that the extension object forwards the request to additional extension objects that are associated with other data objects for processing of the same

request. To read on the claims, Bardasz' operator object must forward the original request to other operators for processing of the original request. Bardasz does not provide for any such forwarding by the operator. The Office Action asserts that the result of an evaluated operator may serve as an argument to other operators. However, regardless of whether an evaluated operator serves as an argument to other operators, Bardasz' operator is not forwarding the original request to other operators. Based on the assertions in the Office Action itself, Bardasz' operator is merely passing the "result" of an operator to other operators. Such a "result" is not equivalent to the claimed request that is forwarded. As claimed, the same request that is received and fulfilled by one extension object is forwarded to other extension objects. Such "request" forwarding as claimed is not even remotely alluded to in Bardasz. Nowhere in Bardasz is there any teaching or description, implicit or explicit, for an operator forwarding a received request to other operators. Instead, an operator performs an action and passes the result to other operators (see col. 12, lines 26-37).

In view of the above, Applicants note that Bardasz provides the ability to generate a dependency graph that depicts a relationship between a data object, relation, and an operator (see col. 6, lines 37-63). Such a teaching does not describe the ability to provide access to data items as set forth in the claims. In this regard, to stretch the interpretation of Bardasz in an attempt to find similarities with the present invention is implausible and beyond the scope of Bardasz and the present invention.

In addition, Applicants note that claim 4 provides additional limitations including super-objects, sub-objects, and data objects. Claim 4 specifically provides that the request concerns such objects. Accordingly, in accordance with claim 1, since the request concerns such super or sub-objects, the request is forwarded to such objects. In rejecting the original claim 4, the Office Action merely ignored the limitations relating to super-objects and sub-objects. Such language has a specific meaning and cannot merely be ignored when rejecting the claims. Accordingly, the Office Action has failed to set forth a prima facie case of obviousness with respect to claim 4.

The various elements of Applicants' claimed invention together provide operational advantages over the systems disclosed in Bardasz and APA. In addition, Applicants' invention solves problems not recognized by Bardasz and APA.

Thus, Applicants submit that independent claims 1, 17, and 20 are allowable over Bardasz and APA. Further, dependent claims 2-6, 8-16, and 18-19 are submitted to be allowable over

Bardasz and APA in the same manner, because they are dependent on independent claims 1, 17, and 20, respectively, and because they contain all the limitations of the independent claims. In addition, dependent claims 2-6, 8-16, and 18-19 recite additional novel elements not shown by Bardasz and APA.

IV. CONCLUSION

In view of the above, it is submitted that this application is now in good order for allowance and such allowance is respectfully solicited. Should the Examiner believe minor matters still remain that can be resolved in a telephone interview, the Examiner is urged to call Applicants' undersigned attorney.

Respectfully submitted,

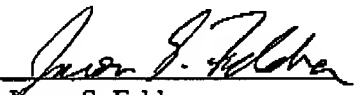
Jirka Stejskal et al.

By their attorneys,

GATES & COOPER LLP

Howard Hughes Center
6701 Center Drive West, Suite 1050
Los Angeles, California 90045
(310) 641-8797

Date: January 13, 2005

By: 
Name: Jason S. Feldmar
Reg. No.: 39,187

JSF/io